NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

COMPUTATIONAL METHODS FOR DETERMINISTIC AND STOCHASTIC NETWORK INTERDICTION PROBLEMS

by

Kelly James Cormican

March, 1995

Thesis Co-Advisors: R. Kevin Wood
David P. Morton

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1995	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE COMPUTATIONAL METHODS FOR DETERMINISTIC AND STOCHASTIC NETWORK INTERDICTION PROBLEMS	5. FUNDING NUMBERS
6. AUTHOR(S) Cormican, Kelly James	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)* Using limited resources, a network interdictor attempts to disable components of a capacitated network with the objective of minimizing the maximum network flow achievable by the network user. This problem has applications to reducing the importation of illegal drugs and planning wartime air attacks against an enemy's supply lines. A deterministic model using Benders decomposition is developed and improved upon with an original "flow-dispersion heuristic." An extension is made to accommodate probabilistic scenarios, where each scenario is an estimate of uncertain arc capacities in the actual network. A unique sequential-approximation algorithm is utilized to investigate cases where interdiction successes are binary random variables.

For a network of 3200 nodes and 6280 arcs, Benders decomposition solves the network interdiction problem in less than one-third of the time required by a direct branch-and-bound method. The flow-dispersion heuristic can decrease solution time to one-fifth or less of that required for the Benders decomposition algorithm alone. With six allowable but uncertain interdictions in a network of 100 nodes and 84 possible interdiction sites among 180 arcs, a stochastic network interdiction problem is solved to optimality in 24 minutes on a IBM RISC/6000 Model 590. With uncertain arc capacities in five scenarios, and three allowable and certain interdictions, a 900 node and 1740 arc network is solved to optimality in 17 minutes on a 60MHZ Pentium PC.

14. SUBJECT TERMS Network, Interdiction, Benders decomposition, Stochastic Programming	15. NUMBER OF PAGES 76
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

Approved for public release; distribution is unlimited.

Accesion For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

COMPUTATIONAL METHODS FOR DETERMINISTIC AND STOCHASTIC NETWORK INTERDICTION PROBLEMS

Kelly James Cormican
Lieutenant, United States Navy
B.S.E., University of Michigan, 1988

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 1995

Author: ______________________
Kelly J. Cormican

Approved by: ______________________
R. Kevin Wood, Co-Advisor

____
David P. Morton, Co-Advisor

____
Gerald G. Brown, Second Reader

____
Peter Purdue, Chairman
Department of Operations Research

ABSTRACT

Using limited resources, a network interdictor attempts to disable components of a capacitated network with the objective of minimizing the maximum network flow achievable by the network user. This problem has applications to reducing the importation of illegal drugs and planning wartime air attacks against an enemy's supply lines. A deterministic model using Benders decomposition is developed and improved upon with an original "flow-dispersion heuristic." An extension is made to accommodate probabilistic scenarios, where each scenario is an estimate of uncertain arc capacities in the actual network. A unique sequential-approximation algorithm is utilized to investigate cases where interdiction successes are binary random variables.

For a network of 3200 nodes and 6280 arcs, Benders decomposition solves the network interdiction problem in less than one-third of the time required by a direct branch-and-bound method. The flow-dispersion heuristic can decrease solution time to one-fifth or less of that required for the Benders decomposition algorithm alone. With six allowable but uncertain interdictions in a network of 100 nodes and 84 possible interdiction sites among 180 arcs, a stochastic network interdiction problem is solved to optimality in 24 minutes on a IBM RISC/6000 Model 590. With uncertain arc capacities in five scenarios, and three allowable and certain interdictions, a 900 node and 1740 arc network is solved to optimality in 17 minutes on a 60MHZ Pentium PC.

THESIS DISCLAIMER

The reader is cautioned that the computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

This thesis develops new mathematical methods for the effective employment of limited resources to reduce the undesirable flow of a commodity that can be moved through a capacitated transportation system by an adversary. Typically, this commodity is moved along routes made up of many shorter interconnected segments, or links, giving the adversary flexibility to select a variety of transportation routes from one or more initial sources to one or more ultimate destinations. If information can be obtained about these routes, a mathematical model of the transportation system can be constructed and represented as a network of junctions and links between the junctions. Each link (and possibly, each junction) has a capacity, i.e., an upper limit on how much of the commodity it can accommodate in a given time period. By expending resources on a link or junction in the network, an interdictor may stop, or interdict, all flow of the commodity on that link or junction. Given a limited budget for resources that can operate on the network, the interdictor can analyze the network to determine the best interdiction locations to achieve the greatest reduction in the flow of the commodity.

This network interdiction problem has applications to curbing the importation of illegal drugs, disrupting an illegitimate communications network, or wartime air attacks against an enemy's supply lines. This problem has been studied before, especially during the Vietnam War effort. These existing works, however, tend to be specific to the application and not easily adaptable to variations and enhancements. More recently developed techniques, while offering many advantages over the methods used previously, may still have difficulty solving a large-scale network interdiction problem in a reasonable period of time.

Using the well-known technique of Benders decomposition, this thesis addresses this shortcoming by developing a solution by separating the network interdiction problem into several smaller problems, which when solved sequentially, solve the original problem. The nature of this decomposition technique allows the observation of both lower and upper bounds on the optimal solution while the problem is being solved.

The decomposition algorithm may be viewed as a sequence of actions and reactions between the interdictor and the adversary. The interdictor can be thought of as reacting to the adversary's rerouting of flow subject to a previous interdiction decision. Therefore, we can enhance the decomposition algorithm by requiring the adversary to maximize flow while simultaneously keeping flow on any individual link or junction as small as possible. This allows the interdictor to gain more information about which links or junctions are the most important, helping to reduce the time required to obtain a solution.

This research also explores the effects of uncertainty on the network interdiction problem. One study of uncertainty considers the implications of links with variable capacity. This study also considers the possibility that a given link is not, in fact, present in the network. Another study employs a unique method to investigate the effect of incorporating interdictions that may fail at certain locations in the network. Either the interdiction attempt is completely successful at stopping flow in that location, or it is entirely unsuccessful.

The usefulness of the decomposition technique is undisputed as a viable alternative for solving large-scale network interdiction problems. Two example cases show dramatic improvements in the time required to obtain a near-optimal solution. The enhancement procedure to aid in decreasing the number of interdictor-adversary actions and reactions also produces, in most cases, further reductions in solution time.

ACKNOWLEDGEMENT

I must express my sincerest gratitude to Professors Kevin Wood and David Morton for their guidance and encouragement while helping me to reach this academic milestone. I would also like to thank Professors Jerry Brown and Siriphong Lawphongpanich for their helpful advice and assistance. Finally, I am deeply grateful to my wife, without whose patience, love, and understanding, I would not have had the freedom to explore this captivating subject.

I. INTRODUCTION

This thesis develops new mathematical programming methods for the effective employment of limited interdiction assets to reduce the flow of a commodity that can be moved through a capacitated transportation system. An adversary strives to maximize flow of the commodity through the system, represented as a network, while an interdictor, with limited assets, attempts to interdict (destroy) arcs or links in the network to minimize the maximum flow. While this thesis is motivated by the ongoing effort to curb the importation of illegal drugs, many other applications of these methods exist. Some other uses may include disrupting an illegitimate communications network or wartime air attacks against an enemy's supply lines. The primary solution methodologies employ Benders decomposition for both deterministic and stochastic models. The stochastic programming models incorporate scenarios and approximation techniques to examine the effects of uncertainty with respect to network topology, capacities, or success of interdiction.

A. BACKGROUND

The familiar war on drugs is a war on two fronts, supply and demand. From the moment anti-drug efforts first became a serious public issue, the debate has raged over which front deserves most attention. Without engaging in this debate, the focus here is on methods to reduce the supply of illegal drugs transported into the United States from abroad. These drugs are typically moved along routes made up of many shorter interconnected segments, giving the drug trafficker flexibility to select the complete transportation route from initial source to ultimate destination. If planners can obtain information about these routes, a mathematical model of the drug transportation system can be constructed and represented as a capacitated network. Given a limited budget for anti-drug resources that operate on the network, e.g., ground inspection teams, surveillance aircraft, etc., planners can analyze the network to determine where best to expend this budget to achieve the greatest reduction in illegal drug flow into the United States.

The network interdiction problem is addressed from the viewpoint of a network

interdictor. From this stance, an interdictor observes a network user striving to move as much commodity as possible from an origination point to a destination point. A network interdictor is also aware of a network user's ability to reroute flow around an interdiction site in the network. Under these conditions, an interdictor works to reduce the undesirable outcome of a user achieving a large flow of commodity through the network. Specifically, a network interdictor attempts to minimize the maximum flow in the network subject to his limited supply of interdiction resources, as each interdiction demands a resource expenditure at the interdiction site. Therefore, an interdictor must make the best decisions possible about where to apply his resources. A network user has no reprisal; he endeavors to produce his best flow results while constrained by the destructive efforts of the interdictor and the capacity of the network.

This problem and many variants have been studied by others under various labels. These range from the very general, for example, "Removing Arcs from a Network" by Wollmer (1964), to the very specific, such as, "Algorithm for Targeting Strikes in a Lines-of-Communications Network," also by Wollmer (1970). Other contributors address similar topics under other titles, but almost all these works share the common characteristic of being specific to the application and not easily generalizable. Two recent works by Steinrauf (1991) and Wood (1993) overcome this limitation by adopting a mathematical programming approach that readily generalizes and is easily adaptable to a variety of network interdiction applications. These mathematical models, however, are difficult-to-solve integer and mixed integer programs. Wood shows that the basic network interdiction problem is NP-complete, even when restricted to planar graphs where interdictions require varying amounts of resource, or to non-planar graphs requiring only one unit of resource per arc.

Advocating the advantages of a generalizable approach, this thesis develops new mathematical programming techniques for the network interdiction problem that offer significant advantages over earlier methods by splitting a difficult-to-solve problem into a sequence of several, more manageable problems that are easier to solve. Finally, and perhaps most significantly, this thesis begins to explore the implications of uncertainty in

network interdiction.

B. NETWORKS AND INTERDICTION

1. Description of a Network

A directed network, denoted G = (N,A), has node set N and arc set A. The total number of nodes and arcs in a network is denoted $|N|$ and $|A|$, respectively. An arc is an ordered pair (i,j) with $i,j \in N$. For an arc (i,j), i is the "tail node" from which the arc originates, and j is the "head node" at which the arc terminates. It is assumed that G contains no arcs of the form (i,i). In a transportation network, an arc (i,j) can be thought of as a length of roadway or a river segment that provides a path for the flow of a commodity from i to j. A model where commodity can flow in either direction on an arc, i.e., arcs are undirected, is discussed later in this chapter in Section B.5. A node i can be thought of as a road junction or waypoint. A commodity flowing through the network originates at a source node $s \in N$ in the network, and flows to a sink node $t \in N$.

Each arc (i,j) has an associated set of parameters that describe its characteristics. The finite nominal capacity, or maximum allowable flow on an arc is denoted u_{ij}, where $u_{ij} \geq 0$. The cost, in units of resource, to interdict an arc (i,j) is r_{ij}, typically assumed to be a small integer. It possible that an arc cannot be interdicted at *any* cost for political, tactical, or other reasons and therefore $r_{ij} = \infty$. A total of R units of resource are available for interdiction. More parameters may be considered, since it is possible to include node capacities and node interdictions, as discussed in Section B.5 of this chapter.

2. Network Maximum Flow Models

The standard maximum flow linear programming model (e.g., Ahuja, et al., 1993, p. 168) determines the maximum quantity of a commodity that can be moved through a capacitated network from source node s to destination node t. This model is

$$\text{MF} \qquad \max_{x} \; x_{ts}$$

$$\text{s.t.} \quad \sum_j x_{sj} - \sum_j x_{js} - x_{ts} = 0$$

$$\sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_j x_{tj} - \sum_j x_{jt} + x_{ts} = 0$$

$$0 \le x_{ij} \le u_{ij} \quad \forall\, (i,j) \in A$$

where x_{ij} is the flow of commodity from node i to node j on directed arc $(i,j) \in A$, and x_{ts} is the flow from sink node t to source node s, on an artificial arc (t,s).

By partitioning the nodes of a network into two sets N_s and N_t, with $s \in N_s$ and $t \in N_t$, the set $\{N_s,N_t\}$ forms an *s-t cutset*. With respect to that cut, an arc is a "forward" arc if it is directed from a node in N_s to a node in N_t. The capacity of the cut is the sum of the capacities of all forward arcs associated with the cut. A minimum cutset, then, is a cutset of minimum capacity among all possible cutsets in the network. By the well-known maximum flow-minimum cut theorem, the maximum flow in a capacitated network is equal to the minimum cutset capacity (Ford and Fulkerson, 1956). A minimum cutset can be found directly by solving the dual of the maximum flow problem (e.g., Wood, 1993):

$$\text{MFD} \qquad \min_{\alpha,\beta} \sum_{(i,j)\in A} u_{ij}\beta_{ij}$$

$$\text{s.t.} \quad \alpha_i - \alpha_j + \beta_{ij} \ge 0 \quad \forall\, (i,j) \in A$$

$$\alpha_t - \alpha_s \ge 1$$

$$\beta_{ij} \ge 0 \quad \forall\, (i,j) \in A.$$

Since MFD is totally unimodular, all variables will be 0 or 1 in an optimal extreme point

solution. The variables in the model have the following physical interpretation: $\alpha_i = 1$ indicates $i \in N_t$, $\alpha_i = 0$ indicates $i \in N_s$, and $\beta_{ij} = 1$ if arc (i,j) is a forward arc of the minimum capacity cut (otherwise $\beta_{ij} = 0$).

3. The Network Interdiction Problem

The network interdiction problem can be formalized in a min-max flow-based model. The network user attempts to maximize the flow across the network, while the interdictor is simultaneously striving to minimize this maximum flow while observing a budget constraint. In this model, $\gamma_{ij} = 1$ if arc (i,j) is interdicted and $\gamma_{ij} = 0$ if the arc is not interdicted. The model is

$$
\begin{aligned}
\textbf{MINMAX} \quad & \min_{\gamma \in \Gamma} \max_{x} \; x_{ts} \\
\text{s.t.} \quad & \sum_j x_{sj} - \sum_j x_{js} - x_{ts} = 0 \\
& \sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\} \\
& \sum_j x_{tj} - \sum_j x_{jt} + x_{ts} = 0 \\
& 0 \le x_{ij} \le u_{ij}(1 - \gamma_{ij}) \quad \forall\, (i,j) \in A
\end{aligned}
$$

where $\Gamma = \{\gamma_{ij} : \gamma_{ij} \in \{0,1\} \;\; \forall\, (i,j) \in A, \sum_{(i,j) \in A} r_{ij}\gamma_{ij} \le R\}$. The resource constraint is for the interdiction of arcs using a single type of resource; this constraint may be expanded to accommodate multiple resource types and other more complicated restrictions. As generalizations of model MINMAX, the models presented in this thesis allow multiple source and sink nodes.

This problem and some variants have been extensively studied in the past in a planar network setting (e.g., Wollmer, 1969), but only recently has a more general mathematical programming approach been applied (Steinrauf, 1991; Wood, 1993). This thesis will extend the mathematical programming approach and explore both deterministic and stochastic networks, and stochastic interdictions.

The approach here is not game-theoretic. In game theory, it is assumed both opponents have the ability to make decisions based on known probabilities that the opponent will take each possible action. In contrast, this thesis makes the assumption that the network interdictor will interdict with impunity and the network user must maximize flow, subject to the interdictions.

Once an arc is interdicted, the network user is assumed to have complete knowledge of the interdiction and reroutes the commodity as best possible. For example, suppose cocaine is being shipped into Miami from Columbia along two major routes: (1) Columbia - Nicaragua - Miami; and (2) Columbia - Jamaica - Nassau- Miami. Suppose each leg along the major routes (1) and (2) can accommodate 10 kg and 20kg of cocaine traffic per month, respectively. With no law enforcement action, 30kg of cocaine per month will arrive in Miami. If the budget allows one interdiction, the best choice is to interdict any leg along major route (2) (i.e., Columbia - Jamaica, Jamaica - Nassau, or Nassau - Miami) to stop 20kg of cocaine per month. The drug runner maximizes drug flow over the remaining network along major route (1), still succeeding in delivering 10kg per month of cocaine to Miami. If law enforcement had interdicted along major route (1) instead, the drug runner could get 20kg of cocaine per month into Miami along the higher capacity connections of major route (2). If it costs law enforcement $1M to interdict major route (1), and $3M to interdict major route (2), the interdiction location must be major route (1) under a $2M budget restriction.

The above example hints at the subtle role that arc capacity can play. In the realm of drug interdiction, a likely value of capacity may be the greatest volume of drug traffic that can be moved on an arc during a given time period without attracting the attention of law enforcement. On the other hand, a value of capacity may be more closely linked to physical assets. For example, only a few vessels or persons may be available to complete an ocean transit at any given time.

4. Basic Network Interdiction Model

Recent work by Steinrauf (1991) and Wood (1993) overcomes many shortcomings of previous models by developing an integer programming model that minimizes the

maximum flow. This thesis expands upon their model, which is:

$$\text{IP1} \qquad \min_{\alpha,\beta,\gamma} \sum_{(i,j)\in A} u_{ij}\beta_{ij}$$

$$\begin{aligned} \text{s.t.} \quad \alpha_i - \alpha_j + \beta_{ij} + \gamma_{ij} &\geq 0 \quad \forall\,(i,j)\in A \\ \alpha_t - \alpha_s &\geq 1 \\ \sum_{(i,j)\in A} r_{ij}\gamma_{ij} &\leq R \\ \alpha_i &\in \{0,1\} \quad \forall\, i\in N \\ \beta_{ij}, \gamma_{ij} &\in \{0,1\} \quad \forall\,(i,j)\in A. \end{aligned}$$

The interpretations of variables in IP1 are similar to those of the dual of the maximum flow formulation MFD, with the addition of the interdiction variable, $\boldsymbol{\gamma}$. With respect to some cutset $\{N_s,N_t\}$, $\alpha_i = 1$ indicates node $i \in N_t$, $\alpha_i = 0$ indicates node $i \in N_s$, $\gamma_{ij} = 1$ if arc (i,j) is a forward arc of the cutset and is interdicted (otherwise $\gamma_{ij} = 0$), and $\beta_{ij} = 1$ if arc (i,j) is a forward arc of the cutset but is not interdicted (otherwise $\beta_{ij} = 0$). As before, the resource constraint is for the interdiction of arcs using a single type of resource, but could be expanded to handle multiple resource types.

5. Undirected Arcs and Interdictable Nodes

For simplicity of presentation, we develop network interdiction models considering only directed arcs and interdictions restricted to arcs. However, since undirected arcs and node interdictions are possible in practice, modified versions of some models accommodating these variations are also discussed, with computational results presented in Chapter II. These variant models use transformations standard in the literature (e.g., Ahuja, et al., 1993, pp. 38-43).

In many cases, including the network interdiction problem, it is necessary to consider arcs that are both directed and undirected. Most roads, for example, allow

transport of a commodity in either direction. To analyze such a network, each undirected arc (i,j) in the original network $G = (N,A)$ is represented by two directed arcs, (i,j) and (j,i) in anti-parallel. Arcs (i,j) and (j,i) would then exist in an equivalent transformed directed network, on which all computations would be performed. An interdiction of arc (i,j) implies an interdiction of arc (j,i), and vice versa. To effect this, the capacity constraint in model MINMAX is replaced by

$$\left.\begin{aligned} 0 \le x_{ij} \le u_{ij}(1 - \gamma_{ij}) \\ 0 \le x_{ji} \le u_{ij}(1 - \gamma_{ij}) \end{aligned}\right\} \quad \forall\,(i,j) \in A.$$

The interdictor, by expending r_{ij} units of resource, stops the flow of commodity in both directions on undirected arc (i,j).

In addition to interdicting arcs, the network interdictor may find node interdiction attractive. This is appealling if node capacities are smaller than or about the same size as arc capacities, or the number of nodes in the network is small relative to the number of arcs. Node interdiction is facilitated by first transforming the network by "node splitting." Such a node i in the original network is split into two nodes i' and i'' in a transformed network, joined by a directed arc from i' to i'' with capacity equal to the capacity of the original node i. All original arcs that were directed toward i are now directed to i', and all arcs originating from i now come exclusively from i''. All such nodes in the original network are then handled as arcs in the transformed network. An interdiction of "arc" (i',i'') is interpreted as an interdiction of node i in the original network.

Admitting undirected arcs and interdictable nodes may provide more realistic modeling, but computations can be hindered by the larger size of the transformed network. In a large, undirected network with all nodes interdictable, it is possible the added computational burden may be substantial.

C. LITERATURE SEARCH

During the Vietnam War, efforts to destroy enemy supply lines produced the first of many studies of the network interdiction problem. The list of contributors is extensive: Wollmer (1964, 1970, 1970), Durbin (1966), McMasters and Mustin (1970), Helmbold (1971), Ghare, Montgomery, and Turner (1971), Lubore, Ratliff, and Sicilia (1971,1975). More recent contributors are Cunningham (1985), Steinrauf (1991), Phillips (1992), and Wood (1993). The recent efforts, inspired mostly by the anti-drug crusade, have attempted to generalize on the earlier, more specific approaches. This section describes some of the more interesting works on the subject.

1. Previous Work

Many works in the literature assume the network in question is "s-t planar." A network is *planar* if it can be drawn in a two-dimensional plane such that no two arcs cross (intersect) each other. A *face* in a planar network is a region of the two-dimensional plane bounded by arcs in which any two points can be connected by a continuous curve that intersects no nodes and no arcs. Finally, an *s-t planar* network is a planar network with source node s and sink node t where both s and t lie on the boundary of the outer face (e.g., Ahuja, et al., 1993, pp. 260-263).

Wollmer (1964) first studied the deterministic network interdiction problem. Wollmer starts by constructing a modified topological dual of an undirected s-t planar network; dual network nodes are located in each face of the original (primal) network and each dual arc crosses an original arc to connect the dual nodes. The length of each dual arc is equal to the capacity of the primal arc that it crosses. Reminding the reader of the maximum flow-minimum cut theorem, Wollmer shows how the problem of finding the minimum cut in the primal network is equivalent to finding the shortest path through the dual network from the dual source to the dual sink. If the resource budget allows n interdictions, the n optimal arcs for interdiction are those arcs that when assigned zero length will minimize the shortest route in the dual network. He then presents a polynomial-time labeling algorithm that accomplishes this by analyzing a modified dual network in which each dual arc is replaced by two parallel arcs. One arc has length zero

and the other has length equal to the capacity of the associated primal arc. The optimal set of arcs to interdict in the primal network then corresponds to the set of zero-length arcs in the shortest path of the modified dual. Unfortunately, the use of the dual network restricts the problem to planar graphs, since otherwise the dual network cannot be constructed. Also, Wollmer assumes that the cost of interdiction does not vary from arc to arc.

Phillips (1992) presents several pseudo-polynomial time algorithms using dynamic programming for interdiction of undirected planar networks. The algorithms are pseudo-polynomial since they allow each interdiction to require a different amount of resource. Phillips also shows how to modify these algorithms to achieve approximations of the optimal solution in polynomial time. A simplifying assumption is made that arcs can be partially interdicted, so that expending a fraction of the cost necessary to interdict an arc removes the corresponding fraction of flow. All computations involve use of the dual of the network, similar to Wollmer. However, there is no requirement here that the network be *s-t* planar. Phillips goes on to describe algorithm modifications to allow complete or partial arc interdiction, and to accommodate a general interdiction function such that the cost of interdiction increases as increasing amounts of interdiction resources are expended. Finally, a proof shows that the basic network interdiction problem is NP-complete.

Steinrauf (1991) and Wood (1993) solve the network interdiction problem with mathematical programming techniques. By avoiding the use of the dual network, the topology of the network is unrestricted. Wood shows how the model is easily generalized to accommodate binary (complete) or continuous (partial) arc interdiction, node interdiction, multiple sources and sinks, undirected networks, multiple resources, and multiple commodities. Wood does point out, however, that as network interdiction problems become larger, certain measures may be necessary to decrease solution times. To this end, two types of valid inequalities are presented to tighten the LP relaxation of the models. Although Wood's computational experimentation does achieve a reduction in solution time, a practical drawback of this method is that the amount of effort needed to construct such inequalities may not justify any time savings gained from their use.

Although several works on this subject exist, they all may be categorized as being not as generalizable as Wood's model, and the models presented in this thesis.

D. PROPOSED MODELS AND SOLUTION TECHNIQUES

Motivated by a need to solve network interdiction problems of increasing size, it is useful to consider techniques other than trying to solve the basic integer program directly, with or without valid inequalities. Having recognized the advantages of the generality of the mathematical programming approach, this thesis begins with the model of Steinrauf and Wood and develops an alternate solution technique using Benders decomposition (Benders, 1962). This algorithm decomposes the network interdiction problem into two problems that are usually much easier to solve. The simpler of the two is a network maximum flow problem, which is solved very quickly. The interdiction decisions are made by an integer program that, at the onset of the algorithm, also produces the optimal solutions to a relaxed problem quickly, since it is small and simple. The optimal solution to the network interdiction problem is constructed by iteratively solving these two problems. The final solution to the network interdiction problem should be rapidly determined if the number of iterations is not too large, since the decomposition algorithm demands that the integer program grow in size by one constraint at each iteration. Therefore, if the number of decomposition algorithm iterations is limited, some time savings may be achieved over directly solving the integer programming model.

With a desire to solve practical problems, network interdiction is also addressed in the stochastic arena. Both simple and more advanced solution techniques are explored to give results under various manifestations of uncertainty. Since the purpose here is to explore various methodologies, and not achieve absolute efficiency, all computations should be regarded as prototypic. The General Algebraic Modeling System (GAMS) (Brooke, et al., 1988) is used to formulate the equations and interface the network data with the algorithms. This software is not specific for this problem, or for decomposition techniques. Consequently, more computation time is required than is actually necessary to solve this problem, if software were designed specifically for this purpose. To obtain solutions to the algorithms formulated with GAMS, the solvers XA (Sunset Software,

1987) and XS (Insight, Inc., 1994) are used. The final stochastic programming algorithm uses more advanced techniques in a sequential-approximation technique (Cormican, Morton, and Wood, 1995). This algorithm is solved with OSL (International Business Machines Corp., 1991).

The algorithms and solution techniques explored in this thesis are:

1. Deterministic Networks with Benders Decomposition

This method employs an iterative solution procedure using Benders decomposition. In the Benders master problem, a set of arcs is chosen for interdiction subject to a budget constraint. The Benders subproblem maximizes flow subject to fixed interdiction locations. The master and subproblems are solved iteratively until the gap between the lower bound from the master problem and the upper bound from the subproblem is small enough to satisfy optimality criteria set by the user.

2. Deterministic Networks with Benders Decomposition and Heuristic

A modification to the straightforward Benders decomposition algorithm, this technique employs a "flow-dispersion" heuristic after solving the Benders subproblem. The heuristic has the effect of dispersing the maximum flow from source(s) to sink(s) throughout the network by approximately solving a minimum cost flow problem with a quadratic cost function that keeps flow on any arc no larger than it has to be to still achieve the maximum flow through the network. The result is more rapid convergence of the upper and lower bounds, decreasing the time and number of algorithm iterations required to produce a satisfactory solution.

3. Simple Stochastic Models with Benders Decomposition

If there is some uncertainty with respect to arc capacities, a set of "scenarios" may be constructed where each scenario is one realization of several possible combinations of arc capacities in the network. Assigned to each scenario is a probability that the scenario represents the actual network, with all arc capacities as specified in the scenario. In this treatment, uncertain capacity also models uncertain network topology since an arc with a capacity of zero models an "arc" that is not present in the actual network. If the number of possible scenarios is manageable, direct integer programming may be used, but as the

number of scenarios grows, Benders decomposition becomes more attractive. The application of Benders decomposition to a set of probabilistic scenarios was developed as the L-shaped algorithm by Van Slyke and Wets (1969).

4. Stochastic Networks with a Sequential-Approximation Algorithm

If the number of probabilistic scenarios becomes too large, the scenario method becomes computationally burdensome since the number of Benders subproblems to solve at each iteration is equal to the number of scenarios. To avoid such difficulties, a sequential-approximation algorithm is employed (Cormican, Morton, and Wood, 1995; Kall, et al., 1988). This algorithm sequentially refines partitions of the sample space and employs the L-shaped algorithm to solve a sequence of approximating problems. This technique can accommodate uncertainty in arc capacities as well as interdiction success, and assumes all random variables are independent. In this thesis, we explore only the situation where uncertainty takes the form of complete success or failure for interdiction attempts. It is possible to extend the methodology to discrete and continuous distributions for interdiction success and arc capacities, as well as certain types of dependency structures for the stochastic parameters, but these topics are beyond the scope of this research.

II. DETERMINISTIC NETWORK INTERDICTION

Solving network interdiction problems on large networks may be computationally impossible with standard integer programming techniques. This possibility motivates the evaluation of decomposition techniques that separate a large complicated problem into a sequence of smaller, quicker-to-solve models. In this chapter, Benders decomposition for the deterministic network interdiction problem is developed, improved upon, and computationally tested.

A. NETWORK INTERDICTION WITH BENDERS DECOMPOSITION

In this section, a decomposed formulation is developed from the integer program formulation and is computationally tested.

1. Model

Benders decomposition is a solution technique that is best applied when a certain set of "complicating" variables link what would otherwise be separate, easily solved models. The network interdiction problem is a prime candidate for Benders decomposition due to the complicating binary interdiction variables: if the interdiction variables are fixed at zero or one for every arc, the resulting model is a network flow problem that is easily solved by several methods. If the choices for interdiction can be made relatively quickly at each iteration, the decomposed problem may offer some advantage over the original integer program model. The derivation of the decomposed model begins with Wood's basic model, which is restated as

$$\text{IP1} \qquad \min_{\gamma,\beta,\alpha} \sum_{(i,j)\in A} u_{ij}\beta_{ij}$$

$$\begin{aligned}
\text{s.t.}\quad \alpha_i - \alpha_j + \beta_{ij} + \gamma_{ij} &\geq 0 \quad \forall\, (i,j) \in A \\
\alpha_t - \alpha_s &\geq 1 \\
\sum_{(i,j)\in A} r_{ij}\gamma_{ij} &\leq R \\
\alpha_i &\in \{0,1\} \;\; \forall\, i \in N \\
\beta_{ij}, \gamma_{ij} &\in \{0,1\} \;\; \forall\, (i,j) \in A.
\end{aligned}$$

For fixed values of γ, the LP relaxation of IP1 is just the dual of a network flow problem with an intrinsically integer solution. The dual of this LP relaxation may be taken to obtain the equivalent problem:

$$\min_{\gamma\in\Gamma} \max_{x} \; x_{ts} - \sum_{(i,j)\in A} \gamma_{ij} x_{ij}$$

$$\begin{aligned}
\text{s.t.}\quad \sum_j x_{sj} - \sum_j x_{js} - x_{ts} &= 0 \\
\sum_j x_{ij} - \sum_j x_{ji} &= 0 \quad \forall\, i \in N - \{s,t\} \\
\sum_j x_{tj} - \sum_j x_{jt} + x_{ts} &= 0 \\
0 \leq x_{ij} &\leq u_{ij} \quad \forall\, (i,j) \in A
\end{aligned}$$

where $\Gamma = \{\gamma_{ij} : \gamma_{ij} \in \{0,1\} \;\forall\, (i,j) \in A, \sum_{(i,j)\in A} r_{ij}\gamma_{ij} \leq R\}$.

Now, enumerate the set X of all the extreme point solutions of the inner maximization and select that solution with minimum value subject to $\gamma \in \Gamma$. This model

may be written as

$$\min_{\gamma \in \Gamma} \max_{x^k \in X} \; x_{ts}^k - \sum_{(i,j) \in A} \gamma_{ij} x_{ij}^k .$$

Equivalently, the min-max problem may be written as the simple minimization:

$$\begin{aligned} \text{MASTER }(X) \qquad & \min_{\gamma \in \Gamma} \; z \\ & \text{s.t.} \quad z \geq x_{ts}^k - \sum_{(i,j) \in A} x_{ij}^k \gamma_{ij} \qquad \forall\, x^k \in X \\ & \qquad \gamma_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \end{aligned}$$

The efficiency of Benders decomposition rests on the ability to solve MASTER(X) with the enumeration of only a fraction of the extreme points X. The formulation with a subset of constraints corresponding to extreme points $X' \subseteq X$ is the Benders master problem, denoted MASTER(X'). Each constraint in the master problem is generated by the Benders subproblem. At each iteration in the algorithm, the solution to the subproblem is checked to see if it violates any constraints in MASTER(X'). If so, X' and MASTER(X') are updated with the solution to the subproblem (an extreme point). This iterative process repeats until the solution to the subproblem is feasible to MASTER(X). At this point, MASTER(X') has enough information to solve the original problem to optimality. Furthermore, each iteration of the master problem produces a non-decreasing lower bound (LB) on the optimal value of the objective function. Similarly, the subproblem will generate an upper bound (UB, not necessarily non-increasing) at each iteration. With this information, the algorithm may be terminated prior to optimality if the gap between the current lower bound and best upper bound is sufficiently small.

The Benders subproblem is

$$\text{SUB}\ (\hat{\gamma}) \qquad \max_{x} \quad x_{ts} - \sum_{(i,j)\in A} x_{ij}\,\hat{\gamma}_{ij}$$

$$\text{s.t.} \quad \sum_{j} x_{sj} - \sum_{j} x_{js} - x_{ts} = 0$$

$$\sum_{j} x_{ij} - \sum_{j} x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_{j} x_{tj} - \sum_{j} x_{jt} + x_{ts} = 0$$

$$0 \le x_{ij} \le u_{ij} \quad \forall\, (i,j) \in A$$

where $\hat{\gamma}$ is the solution to MASTER(X'). The subproblem is simply a variant of the maximum flow formulation that penalizes flow on any arc chosen for interdiction in the master problem. Letting $z(f)$ denote the optimal value of the objective function to the optimization problem f, the algorithm is:

Benders Decomposition Algorithm for Network Interdiction

Input: Network G=(N,A), arc capacities u_{ij}, arc interdiction costs r_{ij}, interdiction budget R, special nodes s and t, convergence tolerance *toler*.

Output: Interdiction vector γ^*, which is the solution within (100 · *toler*)% of optimality.

step 1 Solve maximum flow problem MF for flow values $\boldsymbol{x}^1$;
Let $X' = \{\boldsymbol{x}^1\}$;
Let k = 2;
Let UB = z(MF)

step 2 Solve MASTER(X') for $\hat{\gamma}$;
Let LB = z (MASTER(X'))

step 3 Solve SUB($\hat{\gamma}$) for $\boldsymbol{x}^k$;
Let $X' = X' \cup \{\boldsymbol{x}^k\}$;
If z(SUB) < UB then let UB = z(SUB($\hat{\gamma}$)) and $\gamma^* = \hat{\gamma}$;

step 4 If UB - LB $\le$ LB · *toler* then stop: Interdiction set γ^* is a solution to the network interdiction problem with objective function value within (100 · *toler*)% of the optimal

objective function value.

step 5 Let k = k + 1;
Go to step 2.

2. Computational Results

In the network interdiction problem, there are potentially unlimited numbers of combinations of network size and topology, arc direction, arc and node capacities, arc and node interdiction costs, resource budget, and prohibited interdiction locations. We present only a few test cases using GAMS to construct simple grid networks as hypothetical capacitated transportation networks for our calculations.

Since some coherence in tests is desirable between different solution techniques to allow meaningful comparisons, some of the characteristics listed above remain unaltered in all cases studied. Any exception cases to the following "fixed" characteristics are clearly labeled. Three network sizes are analyzed, as shown in Figure 1. All data used in the test cases is available from the advisors of this thesis.

Small network (NS):	100 nodes, 180 directed arcs, 10 by 10 grid;
Medium network (NM):	900 nodes, 1740 directed arcs, 30 by 30 grid;
Large network (NL):	3200 nodes, 6280 directed arcs, 40 by 80 grid.

Figure 1. Networks for Case Studies

In networks NS and NM, all exterior nodes on one side of the grid are defined as source nodes, while all nodes at the opposite side of the grid are defined as sink nodes. In network NL, the nodes at the 40-node sides of the grid are similarly defined as source and sink nodes. Note that the techniques presented in this thesis do not require source and sink nodes be exterior nodes of a network.

Other network and network interdiction characteristics that remain unchanged throughout all testing are shown in Figure 2.

Arc capacity range: [10,100], uniformly distributed in increments of ten; Node capacity range: [10,200], uniformly distributed in increments of ten; Resource cost of interdiction, each arc: 1 (except where noted); Probability that any arc (node) is available for interdiction: 50%.

Figure 2. Fixed Data Characteristics for Network Interdiction Case Studies

It is desirable to perform all calculations with the same hardware, but the computational demands of the larger problems make this impractical. Consequently, the less demanding cases are solved using a 60 MHZ Pentium PC and the more difficult cases utilize an IBM RISC/6000 Model 590. Since our interest is not particular to any one data set, we present summarized case results: the number of Benders decomposition algorithm iterations performed for convergence, total "clock" time for computations, and total solver time for computations, where the (significant) difference between clock and solver is time required for the operations of the GAMS interface.

A total of seven cases are tested with the Benders decomposition algorithm. The first four cases are performed on network NS, with varying resource levels R. In one of the instances where $R = 3$, approximately one-half of the arcs have interdiction costs $r_{ij} = 2$; all other cases consider only $r_{ij} = 1$. The next two cases use network NM with two values of R. Initial results show that solving network NM with $R > 6$, or solving network NL with only the Benders decomposition algorithm, is not efficient. These cases are solved, however, by other means in subsequent sections of this thesis.

Assuming characteristics outlined in Figure 2, a summary of results for network NS follows in Table 1. The 60 Mhz Pentium PC is used for these calculations. The medium network NM was also examined also using the 60 Mhz Pentium PC. With the Figure 2 assumptions of characteristics, a summary for network NM is in Table 2.

From the data in Tables 1 and 2, it is evident that an increase in computational

difficulty is experienced as the resource budget R increases. The test with $R = 9$ was terminated when the time required to achieve a satisfactory solution was deemed

SMALL NETWORK CASE(NS)	NUMBER OF BENDERS ITERATIONS	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 1$	1	00:00:00.8	00:00:15
$R = 3$	5	00:00:02.5	00:00:44
$R = 3$, r_{ij}=1 or 2	5	00:00:02.6	00:00:44
$R = 6$	8	00:00:04.6	00:01:10

Table 1. Benders Decomposition Algorithm Summary of Results, Small Network. Solving the network interdiction problem on network NS using the Benders decomposition algorithm, with 6 units of interdiction resource, takes 8 iterations of the algorithm, 4.6 seconds of solver time, and 1 minute and 10 seconds of clock time.

excessive. All of this additional time is spent solving the Benders master problem, MASTER(X'). This is intuitive, as the budget constraint is of a 0-1 knapsack variety and the number of possible combinations of feasible interdictions grows exponentially with R.

MEDIUM NETWORK CASE(NM)	NUMBER OF BENDERS ITERATIONS	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 3$	8	00:00:24.2	00:02:59
$R = 6$	16	00:00:38.6	00:10:35
$R = 9$	> 40	> 06:12:00	> 09:00:00

Table 2. Benders Decomposition Algorithm Summary of Results, Medium Network. Solving the network interdiction problem on network NM using the Benders decomposition algorithm, with 9 units of interdiction resource, takes more than 40 iterations of the algorithm, more than 6 hours and 12 minutes of solver time, and more than 9 hours of clock time.

To demonstrate the simple extension of network interdiction problems on undirected networks where node interdiction is also allowed, two cases were studied using network NS', which is network NS modified to have either one-half or all of its arcs

undirected. The results are in Table 3. Interestingly, the number of undirected arcs had very little effect on the time required in this small problem.

SMALL NETWORK CASE(NS')	NUMBER OF BENDERS ITERATIONS	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 3$, 50% undirected arcs	10	00:00:07.6	00:02:08
$R = 3$, 100% undirected arcs	10	00:00:07.7	00:02:08

Table 3. Benders Decomposition Algorithm Summary of Results, Small Network with Directed and Undirected Arcs, Node Interdictions Allowed. Solving the network interdiction problem on network NS' using the Benders decomposition algorithm, with all arcs undirected and 3 units of interdiction resource, takes 10 iterations of the algorithm, 7.7 seconds of solver time, and 2 minutes and 8 seconds of clock time.

B. ALGORITHM IMPROVEMENTS

The objective functions of the Benders master and subproblem must converge in a finite number of iterations (Benders, 1962). Unfortunately, the number of iterations required for convergence may be large. It is useful to consider techniques that decrease solution time by reducing the number of iterations the Benders decomposition algorithm requires for convergence.

1. Flow Dispersion Heuristic

Comparisons of solutions to the maximum flow problem and the interdiction problem suggest that arcs that are capacitated, or nearly capacitated, are interdicted more often than those with little or no flow in the optimal solution to the maximum flow problem. This result supports intuition, and leads to the following argument:

> In solving the master problem, the interdictor may achieve a "better" interdiction by interdicting arcs where x^k_{ij} is large. It is therefore likely that γ_{ij} will be 1 where x^k_{ij} is large. But, the magnitude of x^k_{ij} may be misleading. It may be large, but if arc (i,j) is interdicted, the flow might be able to be rerouted with little or no loss. The network user, consequently, would like x^k_{ij} to be closer to the "true" value necessary to achieve a maximum flow.

To this end, we will require the network user to maximize flow while simultaneously keeping flow on any individual arc as small as possible, in some crude sense. In such a solution, if x^k_{ij} is large, it is large only because it *must* be to achieve a maximum flow.

If the maximum flow problem MF is first solved for the value MAXFLOW = z(MF), the solution to the following nonlinear program will achieve a maximum flow and tend to keep flows on individual arcs as small as possible:

$$\text{NLP} \qquad \min_{x} \sum_{(i,j)\in A} x_{ij}^2$$

$$\text{s.t.} \quad \sum_j x_{sj} - \sum_j x_{js} = \mathit{MAXFLOW}$$

$$\sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_j x_{tj} - \sum_j x_{jt} = -\mathit{MAXFLOW}$$

$$0 \leq x_{ij} \leq u_{ij}(1 - \hat{\gamma}_{ij}) \quad \forall\, (i,j) \in A.$$

Solving this nonlinear program *exactly* is computationally difficult. One exact solution method, simplicial decomposition, has been considered promising but is not tested since a solver employing this method is not available. Since the proposed technique is a heuristic, approximate solutions to NLP may be sufficient to produce a satisfactory flow dispersion. This reduces the time needed to solve NLP. Three approximate solution methods are explored in this thesis:

a. Many-Level Approximation Method

The separable quadratic objective function can be approximated by a sum of piecewise linear functions composed of n linear segments of equal length. This corresponds to the replacement of each original arc with n parallel arcs, each with capacity u_{ij}/n. Each new arc is labeled as a "level," l, from 1 to n, and usage costs increase as the square of the level of the l^{th} arc used. This method very closely approximates a quadratic

function, but has the disadvantage of increasing the number of arcs in the network n-fold.

b. Two-Level Iterative Approximation Method

The quadratic functions can be approximated, with iterative improvements, using a two-segment linear approximation. Given an initial approximation, the resulting flow values are then used to adjust the lengths (capacities) and slopes of the two linear segments to reduce the error between the linear approximation and a true quadratic function. This process is repeated several times for the best approximation. This heuristic requires each original arc to be replaced by only two parallel arcs, each with capacity $u_{ij}/2$, only doubling the number of arcs in the network.

c. Frank-Wolfe Method

Developed as a method to solve quadratic programming problems, the Frank-Wolfe algorithm is widely used in non-linear programming (e.g., Bazaraa and Shetty, 1979, p. 184). It is ideal for implementation of the flow-dispersion heuristic.

Starting with a feasible solution $\bar{\boldsymbol{x}}$ to NLP, the direction of movement for an improved objection function value is $-\nabla \boldsymbol{f}(\bar{\boldsymbol{x}})$, projected onto the feasible region. For NLP, one direction-finding problem is given by the following LP:

$$\text{DFP}(\bar{\boldsymbol{x}}) \qquad \min_{\boldsymbol{x}} \sum_{(i,j)\in A} 2\, \bar{x}_{ij}\, x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{sj} - \sum_j x_{js} = \mathit{MAXFLOW}$$

$$\sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_j x_{tj} - \sum_j x_{jt} = -\mathit{MAXFLOW}$$

$$0 \le x_{ij} \le u_{ij}(1 - \hat{y}_{ij}) \quad \forall\, (i,j) \in A$$

where MAXFLOW = z(MF), as before. If $\bar{\boldsymbol{x}}$ is not optimal, an improving direction is $\boldsymbol{d} = \hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}$, where $\hat{\boldsymbol{x}}$ solves DFP($\bar{\boldsymbol{x}}$). The optimal step size, λ^*, is easily found by

solving

$$\min_{\lambda} \quad f(\lambda \bar{x}_{ij} + (1 - \lambda)\hat{x}_{ij})$$

$$\text{s.t.} \quad 0 \le \lambda \le 1$$

where $f(x) = \sum_{(i,j) \in A} x_{ij}^2$. Differentiating with respect to λ and setting the result equal to zero, the optimal step size is

$$\lambda^* = \frac{\sum_{(i,j) \in A} \hat{x}_{ij}^2 - \sum_{(i,j) \in A} \hat{x}_{ij}\bar{x}_{ij}}{\sum_{(i,j) \in A} \hat{x}_{ij}^2 + \sum_{(i,j) \in A} \bar{x}_{ij}^2 - 2\sum_{(i,j) \in A} \hat{x}_{ij}\bar{x}_{ij}} .$$

Because it may be impractical to find x^* exactly, we use a stopping rule to obtain a solution sufficiently close to x^*. The number of times $\hat{x}$ and λ^* must be determined will depend on this stopping rule. Certainly, $f(\bar{x}) \ge f(x^*)$ since $\bar{x}$ is a feasible solution to DFP($\bar{x}$). We know $f(x^*) \ge f(\bar{x}) + \nabla f(\bar{x})(x^* - \bar{x})$, and since $\hat{x}$ minimizes DFP($\bar{x}$), it follows that $f(x^*) \ge f(\bar{x}) + \nabla f(\bar{x})(\hat{x} - \bar{x})$. As a result we can employ a relative stopping rule by checking the magnitude of the relative gap between $\hat{x}$ and $\bar{x}$. The algorithm for this method is now apparent as:

Frank-Wolfe Algorithm for Heuristic
Inputs: Network G=(N,A), current arc flows $\bar{x}$, relative gap ϵ.
Output: Adjusted arc flows $\bar{x}$.

step 1 Solve DFP($\bar{x}$) for $\hat{x}$;

$$\text{If} \left[\frac{2 \sum_{(i,j) \in A} \bar{x}_{ij}^{\,2} - 2 \sum_{(i,j) \in A} \bar{x}_{ij} \hat{x}_{ij}}{2 \sum_{(i,j) \in A} \bar{x}_{ij} \hat{x}_{ij} - \sum_{(i,j) \in A} \bar{x}_{ij}^{\,2}} \right] < \epsilon \text{, then stop}$$

step 2 Compute λ^*

step 3 Update $\bar{x}$: $\bar{x} \leftarrow \lambda^* \bar{x} + (1 - \lambda^*) \hat{x}$;
Go to step 1.

This algorithm is a variant of the Steepest Descent Method, and as a first-order nonlinear method tends to stall, or converge poorly, as the optimal solution is approached (e.g., Bazaraa and Shetty, 1979, p. 290). This may not be a significant problem, however, since it is not necessary to solve the problem exactly.

2. Computational Results

Several cases are tested with the Benders decomposition algorithm and the different implementations of the flow-dispersion heuristic. The final cases analyze network NL with the most efficient heuristic technique and compare the results with those obtained from direct integer programming. To allow comparison, all data for the cases tested here are the same as those in Section A, where the heuristic is not used.

Table 4 presents a summary of results for network NS with the many-level approximation method. Also given is the percent of time or iterations the algorithm required compared to the straightforward Benders decomposition algorithm without the heuristic as tested in Section A of this chapter.

SMALL NETWORK CASE(NS)	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 1$	1	100%	00:00:02.3	288%	00:00:22
$R = 3$	2	40%	00:00:04.1	164%	00:00:35
$R = 3$, r_{ij}=1 or 2	4	80%	00:00:07.8	300%	00:01:05
$R = 6$	16	200%	00:00:38.6	839%	00:04:22

Table 4. Benders Decomposition Algorithm with Heuristic: Many-Level Approximation Method Summary of Results, Small Network. Solving the network interdiction problem on network NS using the Benders decomposition algorithm with the many-level approximation implementation of the flow-dispersion heuristic, with 3 units of interdiction resource, took 2 iterations of the algorithm, 4.1 seconds of solver time, and 35 seconds of clock time. This is 40% of the number of iterations and 164% of the solver time required to solve the problem with Benders decomposition alone.

Using the two-level iterative approximation method, the results are in Table 5.

SMALL NETWORK CASE(NS)	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 1$	1	100%	00:00:01.2	150%	00:00:19
$R = 3$	3	60%	00:00:04.7	188%	00:00:56
$R = 3$, r_{ij}=1 or 2	3	60%	00:00:05.6	215%	00:01:02
$R = 6$	13	163%	00:00:30.4	661%	00:04:20

Table 5. Benders Decomposition Algorithm with Heuristic: Two-Level Iterative Approximation Method Summary of Results, Small Network. Solving the network interdiction problem on network NS using the Benders decomposition algorithm with the two-level iterative approximation implementation of the flow-dispersion heuristic, with 3 units of interdiction resource, took 3 iterations of the algorithm, 4.7 seconds of solver time, and 56 seconds of clock time. This is 60% of the number of iterations and 188% of the solver time required to solve the problem with Benders decomposition alone.

The Frank-Wolfe method produced the results in Table 6. In this instance, the Frank-Wolfe implementation of the heuristic seems to be most effective in the case where differing arc interdiction costs are present.

SMALL NETWORK CASE(NS)	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 1$	1	100%	00:00:01.0	125%	00:00:19
$R = 3$	3	60%	00:00:06.2	248%	00:01:18
$R = 3$, r_{ij}=1 or 2	2	40%	00:00:03.6	138%	00:00:46
$R = 6$	14	175%	00:00:46.2	1004%	00:07:43

Table 6. Benders Decomposition Algorithm with Heuristic: Frank-Wolfe Method Summary of Results, Small Network. Solving the network interdiction problem on network NS using the Benders decomposition algorithm with the Frank-Wolfe implementation of the flow-dispersion heuristic, with 3 units of interdiction resource, took 3 iterations of the algorithm, 6.2 seconds of solver time, and 1 minute and 18 seconds of clock time. This is 60% of the number of iterations and 248% of the solver time required to solve the problem with Benders decomposition alone.

The medium network NM is also examined using a 60 Mhz Pentium PC. Table 7 shows a summary for network NM using the many-level approximation method with the standard assumptions of characteristics. Due to the poor performance of the case where $R = 9$ in Table 2, no test is performed in Section A for a case where $R = 12$ using only the Benders decomposition algorithm. The case where $R = 9$ is a striking demonstration of the usefulness of the flow-dispersion heuristic. With the exception of the case where $R = 6$ using network NS, the heuristic has been successful in reducing the number of iterations of the algorithm, but is unsuccessful at improving solver time. The $R = 9$ case in Table 7, however, clearly shows that it is possible to significantly decrease the solver time by using the heuristic.

MEDIUM NETWORK CASE(NM)	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 3$	2	25%	00:00:42.3	175%	00:03:10
$R = 6$	3	19%	00:01:00.3	156%	00:04:31
$R = 9$	7	< 18%	00:02:26.9	< 1%	00:10:24
$R = 12$	15	NA	00:27:31.6	NA	00:44:31

Table 7. Benders Decomposition Algorithm with Heuristic: Many-Level Approximation Method Summary of Results, Medium Network. Solving the network interdiction problem on network NM using the Benders decomposition algorithm with the many-level approximation implementation of the flow-dispersion heuristic, with 9 units of interdiction resource, takes 7 iterations of the algorithm, 2 minutes and 26.9 seconds of solver time, and 10 minutes and 24 seconds of clock time. This is less than 18% of the number of iterations and less than 1% of the solver time required to solve the problem with Benders decomposition alone. No comparison is made for the test case where $R = 12$.

The two-level iterative approximation method results are in Table 8 and the results using the Frank-Wolfe method are in Table 9. Now, the results in Table 4 through Table 9 show that the heuristic is almost always useful for decreasing the required number of Benders iterations. Unfortunately, this is not always accompanied by a decrease in solver times. This is due to running the heuristic at each iteration. With only a few test cases here, it is not possible to state conclusively that the heuristic has either a positive or negative effect on required solver time, but there is some indication that it may be more effective on the larger and more difficult problems.

MEDIUM NETWORK CASE(NM)	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 3$	2	25%	00:00:23.3	96%	00:02:35
$R = 6$	3	19%	00:00:39.2	102%	00:04:12
$R = 9$	20	< 50%	00:43:59.6	< 12%	03:41:10
$R = 12$	31	NA	01:48:38.0	NA	04:03:40

Table 8. Benders Decomposition Algorithm with Heuristic: Two-Level Iterative Approximation Method Summary of Results, Medium Network. Solving the network interdiction problem on network NM using the Benders decomposition algorithm with the two-level iterative approximation implementation of the flow-dispersion heuristic, with 9 units of interdiction resource, takes 20 iterations of the algorithm, 43 minutes and 59.6 seconds of solver time, and 3 hours, 41 minutes, and 10 seconds of clock time. This is less than 50% of the number of iterations and less than 12% of the solver time required to solve the problem with Benders decomposition alone. No comparison is made for the test case where $R = 12$.

MEDIUM NETWORK CASE(NM)	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 3$	3	38%	00:00:49.1	203%	00:10:53
$R = 6$	5	31%	00:02:19.1	360%	00:23:07
$R = 9$	12	< 30%	00:08:53.6	< 3%	01:23:04
$R = 12$	28	NA	01:35:43.5	NA	04:39:16

Table 9. Benders Decomposition Algorithm with Heuristic: Frank-Wolfe Method Summary of Results, Medium Network. Solving the network interdiction problem on network NM using the Benders decomposition algorithm with the Frank-Wolfe implementation of the flow-dispersion heuristic, with 9 units of interdiction resource, takes 12 iterations of the algorithm, 8 minutes and 53.6 seconds of solver time, and 1 hour, 23 minutes, and 4 seconds of clock time. This is less than 30% of the number of iterations and less than 3% of the solver time required to solve the problem with Benders decomposition alone. No comparison is made for the test case where $R = 12$.

The best method for implementation of the heuristic, as problems become larger, seems to be the many level approximation method. It is significantly more effective for solving the larger, harder problems. For the cases studied, the number of iterations of the Benders decomposition is significantly less for this method compared to the others, about half for the case where $R = 12$. Similarly, the solver time required for the many-level approximation method is only about 20% of either the two-level iterative approximation method or the Frank-Wolfe method. An exception to this statement is the case where $R = 3$ and arc interdiction costs are varied, where the Frank-Wolfe method is the most efficient. The two-level iterative approximation method is also competitive when resource levels are small. If the many-level approximation method is indeed the best, it still has at least one drawback: it requires a larger computer memory capacity than the other methods.

The large network NL is analyzed using the RISC/6000 and the many-level approximation method. The results for network NL are in Table 10.

LARGE NETWORK CASE(NL)	NUMBER OF BENDERS ITERATIONS	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 12$	21	00:36:57.3	05:44:20

Table 10. Benders Decomposition Algorithm with Heuristic: Many-Level Approximation Method Summary of Results, Large Network. Solving the network interdiction problem on network NL using the Benders decomposition algorithm with the many-level approximation implementation of the flow-dispersion heuristic, with 12 units of interdiction resource, takes 21 iterations of the algorithm, 36 minutes and 57.3 seconds of solver time, and 5 hours, 44 minutes, and 20 seconds of clock time. These results are later compared to results from solving this problem directly.

The advantage of the many-level approximation method is made clearer from the results in Table 10. Only 21 iterations of the algorithm are required for the large network, compared with 31 and 28 iterations to solve the medium network using the two-level approximation method and the Frank-Wolfe method, respectively. Comparison of solver and clock times yield similar differences. The large clock time in Table 10 is discouraging,

but the prototypical nature of this research admits such. In practical application, an efficient compiled high-level computer language should be used with solvers rather than prototypic GAMS, which is easy to apply but painfully slow to execute. Also, additional methods beyond the scope of this thesis are available to speed up solution time in the more time-consuming Benders master program.

The many-level implementation of the heuristic is extended to accommodate directed and undirected arcs and node interdictions. The same two cases are studied, using network NS' as before. The results are displayed in Table 11.

SMALL NETWORK CASE(NS')	BENDERS ITERATIONS		SOLVER TIME (hr:min:sec)		TOTAL CLOCK TIME(hr:min:sec)
	NUMBER OF	% OF SEC. A	TOTAL	% OF SEC. A	
$R = 3$, 50% undirected arcs	5	50%	00:00:14.3	188%	00:06:03
$R = 3$, 100% undirected arcs	5	50%	00:00:15.6	203%	00:06:11

Table 11. Benders Decomposition Algorithm with Heuristic: Many-Level Approximation Method Summary of Results, Small Network with Directed and Undirected Arcs, Node Interdictions Allowed. Solving the network interdiction problem on network NS' using the Benders decomposition algorithm with the many-level approximation implementation of the flow-dispersion heuristic, with all arcs undirected and 3 units of interdiction resource, takes 5 iterations of the algorithm, 15.6 seconds of solver time, and 6 minutes and 11 seconds of clock time. This is 50% of the number of iterations and 203% of the solver time required to solve the problem with Benders decomposition alone.

As in the computations of Section A, where the heuristic was not used, the computational effort required depends very little on the number of undirected versus directed arcs in this small problem. We conjecture, however, that this might not be the case when solving a large-scale problem.

Having obtained a sufficient sample of results from the implementation of the Benders decomposition algorithm, with and without the flow-dispersion heuristic, it is

interesting to finally compare these results with those obtained from solving the network interdiction problem directly with model IP1. We suspect any payoff from the decomposition methods shown here will be most evident when solving larger problems. Therefore, we solve the $R = 12$ case with IP1 and present the results in Table 12. The solver terminated prior to reaching a satisfactory solution due to exceeding pre-set time limits. The results are sufficient to clearly demonstrate that, in this example, the Benders decomposition algorithm with the flow-dispersion heuristic provides a solution more quickly than solving the integer program directly. Comparing Tables 10 and 12, we see that we achieve *at least* a 68% reduction in required solver time over the direct solution method. We do not compare clock times because, as previously stated, they include the relatively slow operations of GAMS.

LARGE NETWORK CASE(NL)	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 12$	> 01:57:59	> 03:09:27

Table 12. Integer Programming Summary of Results, Large Network. Solving the network interdiction problem on network NL directly using integer programming model IP1, with 12 units of interdiction resource, takes more than 1 hour, 57 minutes, and 59 seconds of solver time, and more than 3 hours, 9 minutes, and 27 seconds of clock time. The same problem is solved in less than 68% of this solver time using Benders decomposition with the many-level approximation implementation of the heuristic, as shown in Table 10.

III. NETWORK INTERDICTION OF PROBABILISTIC SCENARIOS

Some degree of uncertainty is likely in network interdiction problems. Only the most intensely studied or simplest drug transportation networks will be known in sufficient detail for a planner to be confident that the best interdiction sites can be selected using (only) deterministic methods. More likely, there will be varying degrees of uncertainty associated with arc capacities, node capacities, the very existence of network components, and the success of each interdiction attempt.

Unfortunately, the complexity of this problem can be overwhelming. For example, consider the case of each arc having only two possible values of capacity. If arc capacities are probabilistically independent, then there are $2^{|A|}$ realizations of this network to analyze for the optimal interdiction set. For all but the smallest network, this problem is intractable using standard methods. If, however, a relatively small set of the most likely instances of the network can be constructed and probabilities ascribed to these instances, the problem may be readily solved.

This chapter explores an application of the L-shaped algorithm (Van Slyke and Wets, 1969) to the stochastic network interdiction problem and gives examples of computations. For additional information on general stochastic programming techniques, the interested reader is referred to Kall and Wallace (1994).

A. INTEGER PROGRAMMING METHOD

Suppose the capacity of each component in a network is a discrete random variable. This includes situations where there is a non-zero probability that an arc in the mathematical network *model* may not actually exist in the *real* network. Such a "nonexistent" arc is treated as an actual arc with zero capacity. If this sort of uncertainty is present, a set of "scenarios" may be constructed, where each scenario is one realization of the network in which exactly one instance of each probabilistic arc in the network is selected. By enumeration of all scenarios, the interdiction decision can be made in an optimal manner over all possible network realizations. The interdictor's objective is to minimize the expected value of the adversary's maximum flow.

1. Model

In this model, the capacities of some or all of the arcs are discrete random variables, possibly dependent. A manageable number of scenarios $v \in V$ is established, each occurring with a probability p_v, that describes the possible outcomes of the random variables. For illustration, suppose three separate law enforcement or anti-drug agencies have constructed three differing estimates of monthly cocaine traffic being transported from Jamaica to Nassau by air. The first two agencies estimate 10kg and 20kg, respectively. The third agency, though, believes this particular air route is not used for drug trafficking, i.e., 0kg of monthly cocaine traffic (a nonexistent arc). Confidence in these estimates is ranked and probabilities are assigned in accordance with the confidence ranking, the higher the confidence the higher the probability assignment. Thus the probabilities p_v of 0.35, 0.40, and 0.25 might be assigned to weight the estimates, respectively. Note that in each of the three scenarios in this example, only the Jamaica - Nassau link is addressed; the remainder of the network must be identical between scenarios to allow a proper assignment of probabilities.

Let x_{ijv} be the flow on arc (i,j) in scenario v, and let u_{ijv} be the corresponding arc capacity. A stochastic network interdiction model with $|V|$ scenarios is

$$\min_{\gamma \in \Gamma} \left\{ \sum_{v \in V} p_v \max_{x_v \in X_v} x_{tsv} \right\}$$

where

$$X_v = \left\{ x_{ijv} : \begin{array}{ll} \sum_j x_{sjv} - \sum_j x_{jsv} - x_{tsv} = 0 & \\ \sum_j x_{ijv} - \sum_j x_{jiv} = 0 & \forall\, i \in N - \{s,t\} \\ \sum_j x_{tjv} - \sum_j x_{jtv} + x_{tsv} = 0 & \\ 0 \le x_{ijv} \le u_{ijv}(1 - \gamma_{ij}) & \forall\, (i,j) \in A \end{array} \right\}$$

and $\Gamma = \{ \gamma_{ij} : \gamma_{ij} \in \{0,1\} \ \forall \ (i,j) \in A, \sum_{(i,j) \in A} r_{ij}\gamma_{ij} \le R\}$. This model is simply an extension to the one-scenario (deterministic) model of Wood. Fixing γ_{ij} and taking the dual of the inner maximization yields

$$\min_{\gamma \ \in \ \Gamma} \left\{ \sum_{v \in V} p_v \ \min_{\alpha,\theta} \sum_{(i,j) \in A} u_{ijv}(1 - \gamma_{ij})\theta_{ijv} \right\}$$

$$\left.\begin{array}{rl} \text{s.t.} & \alpha_{iv} - \alpha_{jv} + \theta_{ijv} \ge 0 \ \ \forall \ (i,j) \in A \\ & \alpha_{tv} - \alpha_{sv} \ge 1 \\ & \alpha_{iv} \in \{0,1\} \ \forall \ i \in N \\ & \theta_{ijv}, \gamma_{ij} \in \{0,1\} \ \forall \ (i,j) \in A \end{array}\right\} \ \forall \ v \in V.$$

The model is then linearized by replacing $(1 - \gamma_{ij})\theta_{ijv}$ with β_{ijv} where $\beta_{ijv} \in \{0,1\}$ and $\beta_{ijv} \ge \theta_{ijv} - \gamma_{ij}$ for each scenario v. The resulting model is

$$\min_{\alpha,\beta,\theta,\gamma \in \Gamma} \ \sum_{v \in V} p_v \sum_{(i,j) \in A} u_{ijv}\beta_{ijv}$$

$$\left.\begin{array}{rl} \text{s.t.} & \alpha_{iv} - \alpha_{jv} + \theta_{ijv} \ge 0 \ \ \forall \ (i,j) \in A \\ & \alpha_{tv} - \alpha_{sv} \ge 1 \\ & \beta_{ijv} + \gamma_{ij} - \theta_{ijv} \ge 0 \ \ \forall \ (i,j) \in A \\ & \alpha_{iv} \in \{0,1\} \ \forall \ i \in N \\ & \beta_{ijv}, \theta_{ijv}, \gamma_{ij} \in \{0,1\} \ \forall \ (i,j) \in A \end{array}\right\} \ \forall \ v \in V.$$

Using an argument analogous to that of Wood, the constraints $\beta_{ijv} + \gamma_{ij} - \theta_{ijv} \ge 0$ may be replaced with equalities $\beta_{ijv} + \gamma_{ij} - \theta_{ijv} = 0$ which allows θ_{ijv} to be replaced by $\beta_{ijv} + \gamma_{ij}$. The interdiction set Γ is now stated explicitly as a constraint, yielding the final model:

$$\text{IPV} \qquad \min_{\alpha,\beta,\gamma} \sum_{v \in V} p_v \sum_{(i,j) \in A} u_{ijv} \beta_{ijv}$$

$$\text{s.t.} \qquad \sum_{(i,j) \in A} r_{ij} \gamma_{ij} \leq R$$

$$\left.\begin{aligned} \alpha_{iv} - \alpha_{jv} + \beta_{ijv} + \gamma_{ij} &\geq 0 \quad \forall\ (i,j) \in A \\ \alpha_{tv} - \alpha_{sv} &\geq 1 \\ \alpha_{iv} &\in \{0,1\} \ \forall\ i \in N \\ \beta_{ijv}, \gamma_{ij} &\in \{0,1\} \ \forall\ (i,j) \in A \end{aligned}\right\} \ \forall\ v \in V.$$

Interpretation of variables in IPV is the same within each scenario as for the case where $|V| = 1$, presented earlier as model IP1.

A potential limitation of IPV is the very large number of constraints required for a problem with many scenarios. The computational difficulties imposed by an very large integer program invites the use of decomposition methods of the previous chapter for this multi-scenario situation. We will first, however, examine direct solution of model IPV.

2. Computational Results

A five-scenario example is constructed and solved using network NM and the branch-and-bound implementation in GAMS/XA. This test case confirms that this is an inefficient technique for larger networks with many scenarios; it fails to exploit the special structure of the problem. The results are in Table 13. As in the one-scenario integer programming case tested in Section II.2 (Table 12), the solver terminated prior to reaching a satisfactory solution due to exceeding pre-set time limits. These results will be compared against the Benders decomposition approach, analyzed in the next section.

MEDIUM NETWORK CASE(NM)	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 3$	> 00:47:03	> 01:12:28

Table 13. Integer Programming for Five Scenarios Summary of Results, Medium Network. Solving the network interdiction problem for five scenarios on network NM directly using integer programming model IPV, with 3 units of interdiction resource, takes more than 47 minutes and 3 seconds of solver time, and more than 1 hour, 12 minutes, and 28 seconds of clock time. These results are later compared to results from solving the five-scenario problem with the L-shaped algorithm.

B. BENDERS DECOMPOSITION: THE L-SHAPED ALGORITHM

In the same manner as the one-scenario case, an equivalent formulation is derived from the original integer program that suggests a Benders decomposition algorithm. The final decomposed model and computational results are then presented.

1. Model

Proceeding as before, if γ is fixed, the LP relaxation of IPV is the dual of a network flow problem with an intrinsically integer solution. The dual is taken with respect to α and β and the resource constraint is again represented as an interdiction set Γ. The model becomes

$$\min_{\gamma \in \Gamma} \max_{x} \sum_{v \in V} x_{tsv} - \sum_{v \in V} \sum_{(i,j) \in A} \gamma_{ij} x_{ijv}$$

$$\text{s.t.} \left.\begin{aligned} \sum_j x_{sjv} - \sum_j x_{jsv} - x_{tsv} &= 0 \\ \sum_j x_{ijv} - \sum_j x_{jiv} &= 0 \quad \forall\, i \in N - \{s,t\} \\ \sum_j x_{tjv} - \sum_j x_{jtv} + x_{tsv} &= 0 \\ 0 \le x_{ijv} &\le p_v u_{ijv} \quad \forall\, (i,j) \in A \end{aligned}\right\} \forall\, v \in V.$$

Now, as before, we enumerate all the extreme point solutions of the inner maximization

and then select that solution with maximum value subject to $\gamma \in \Gamma$. The Benders master problem becomes

$$\min_{\gamma \in \Gamma} \ z$$

$$\text{s.t.} \quad z \geq \sum_{v \in V} x_{tsv}^{k} - \sum_{v \in V} \sum_{(i,j) \in A} x_{ijv}^{k} \gamma_{ij} \quad \forall \ x^{k} \in X'$$

$$\gamma_{ij} \in \{0,1\} \quad \forall \ (i,j) \in A$$

where X is the set of all extreme point solutions x^k and $X' \subseteq X$. With $\hat{\gamma}_{ij}$ fixed at 0 or 1, the Benders subproblem for the multi-scenario case is

$$\max_{x} \ \sum_{v \in V} x_{tsv} - \sum_{v \in V} \sum_{(i,j) \in A} x_{ijv} \hat{\gamma}_{ij}$$

$$\left.\begin{aligned} \text{s.t.} \quad \sum_{j} x_{sjv} - \sum_{j} x_{jsv} - x_{tsv} &= 0 \\ \sum_{j} x_{ijv} - \sum_{j} x_{jiv} &= 0 \quad \forall \ i \in N - \{s,t\} \\ \sum_{j} x_{tjv} - \sum_{j} x_{jtv} + x_{tsv} &= 0 \\ 0 \leq x_{ijv} \leq p_v u_{ijv} & \quad \forall \ (i,j) \in A \end{aligned}\right\} \ \forall \ v \in V.$$

The objective function of the subproblem may be rewritten as

$$\max_{x} \ \sum_{v \in V} \{ x_{tsv} - \sum_{(i,j) \in A} \hat{\gamma}_{ij} x_{ijv} \}$$

and the constraints may be separated into $|V|$ independent maximizations, each representative of a different scenario. The objective function value of the complete subproblem is then obtained by summing over all the scenarios. Therefore, the flow variables $\tilde{x}_{ij}$ returned to the master problem as extreme point solutions are sums of the flows in each arc (i,j) over all the scenarios. The master problem simplifies to

$$\min_{\gamma \in \Gamma} \; z$$

$$\text{s.t.} \quad z \geq \tilde{x}_{ts}^{k} - \sum_{(i,j) \in A} \gamma_{ij} \tilde{x}_{ij}^{k}$$

$$\gamma_{ij} \in \{0,1\} \quad \forall\, (i,j) \in A$$

where $\tilde{x}_{ts}^{k} = \sum_{v \in V} x_{tsv}^{k}$ and $\tilde{x}_{ij}^{k} = \sum_{v \in V} x_{ijv}^{k}$. Equivalently, the arc capacity constraint in the subproblem can be replaced by

$$0 \leq x_{ijv} \leq u_{ijv} \quad \forall\, (i,j) \in A$$

in each scenario v. The probability of the occurrence of each scenario, p_v, is taken into account in the aggregation of scenarios: $\tilde{x}_{ts}^{k} = \sum_{v \in V} p_v x_{tsv}^{k}$ and $\tilde{x}_{ij}^{k} = \sum_{v \in V} p_v x_{ijv}^{k}$. Regardless of method, the flow values returned to the master problem as cuts (constraints) are the expected values of flow on each arc, observed over all scenarios. Therefore, each iteration of the algorithm requires the solution of one master problem and $|V|$ subproblems. The arc flows computed in the subproblems are combined in a sum weighted by p_v prior to the next iteration of the master problem.

To summarize this procedure, we display the algorithm below. As before, let $z(f)$ denote the optimal value of the objective function to the optimization problem f.

The L-Shaped Algorithm for Network Interdiction

Input: Network $G=(N,A)$, arc interdiction costs r_{ij}, interdiction budget R, special nodes s and t, convergence tolerance *toler*, and for each scenario v: probability of occurrence p_v and arc capacities u_{ijv}.

Output: Interdiction vector γ^*, which is the solution within $(100 \cdot toler)$% of optimality.

step 1 Let $u_{ij} = \sum_{v \in V} p_v u_{ijv}$;

Solve maximum flow problem MF for flow values x^1;

Let $X' = \{x^1\}$;

Let k = 2;

Let UB = z(MF)

step 2 Solve MASTER(X') for $\hat{\gamma}$;
Let LB = z (MASTER(X'))

step 3 For $v = 1$ to V
Solve $\mathrm{SUB}_v(\hat{\gamma})$ for $\boldsymbol{x}_v^k$
next v;
Calculate $\tilde{\boldsymbol{x}}^k = \sum_{v \in V} p_v \boldsymbol{x}_v^k$;
Let $X' = X' \cup \{\tilde{\boldsymbol{x}}^k\}$;
Calculate $z(\mathrm{SUB}) = \sum_{v \in V} p_v z(\mathrm{SUB}_v)$;
If z(SUB) < UB then let UB = z(SUB) and $\gamma^* = \hat{\gamma}$

step 4 If UB - LB $\leq$ LB $\cdot$ *toler* then stop: Interdiction set γ^* is a solution to the network interdiction problem with objective function value within (100 $\cdot$ *toler*)% of the optimal objective function value.

step 5 Let k = k + 1;
Go to step 2.

Note that the upper bound obtained in step 1 is valid due to Jensen's inequality (e.g., Kall and Wallace, 1994, p. 168), although it may not be a strong bound since there are no interdictions.

2. Computational Results

The same five-scenario example from Section A is analyzed here. The results are presented in Table 14. Comparing the direct integer programming results in Table 13 and the L-shaped algorithm results in Table 14, the advantages of decomposition are clear. Here, we achieve *at least* a 93% reduction in required solver time over the direct solution method. As the number of scenarios, network size, and resource budget grows, the possible advantages of a decomposition approach become more apparent. This result is paralleled earlier for the one-scenario case in Section B.2 of Chapter II. Again, clock times are not compared here due to the relatively slow operations of GAMS.

MEDIUM NETWORK CASE(NM)	NUMBER OF ALGORITHM ITERATIONS	TOTAL SOLVER TIME (hr:min:sec)	TOTAL CLOCK TIME(hr:min:sec)
$R = 3$	13	00:02:57.2	00:17:15

Table 14. L-Shaped Algorithm for Five Scenarios, Summary of Results, Medium Network. Solving the network interdiction problem for five scenarios on network NM using the L-shaped algorithm, with 3 units of interdiction resource, takes 13 iterations of the algorithm, 2 minutes and 57.2 seconds of solver time, and 17 minutes and 15 seconds of clock time. The solver time is a reduction of at least 93% over the solver time required to solve the five scenario problem directly with integer programming, as shown in Table 13.

IV. STOCHASTIC NETWORK INTERDICTION

The multi-scenario approach of the previous chapter is adequate when the number of scenarios is limited. If uncertainty is prevalent in the network or the assumption of dependency between arc capacities is not valid, more efficient means are available to arrive at a solution. In this chapter, a unique algorithm is used for computations. The results are compared to those of the deterministic models.

A. A SEQUENTIAL-APPROXIMATION ALGORITHM

In this chapter, a proposition is stated that aids in the derivation of the stochastic network interdiction problem. An algorithm is presented that efficiently solves the two-stage stochastic program with recourse. Two example problems are solved and discussed.

1. Development

Consider a network with a special subset of arcs $A^* \subseteq A$. If the arcs in A^* have been interdicted, evaluating the maximum flow in the "residual network" can be done by solving M1:

$$\textbf{M1} \qquad \max_{x} \quad x_{ts}$$

$$\text{s.t.} \quad \sum_j x_{sj} - \sum_j x_{js} - x_{ts} = 0$$

$$\sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_j x_{tj} - \sum_j x_{jt} + x_{ts} = 0$$

$$0 \le x_{ij} \le u_{ij} \quad \forall\, (i,j) \in A$$

$$x_{ij} \le 0 \quad \forall\, (i,j) \in A^*.$$

The determination of the set A^* is exogenous to M1. In the development of the Benders decomposition model in Chapter II, it was shown how SUB($\hat{\gamma}$) solves the network interdiction problem with $\hat{\gamma} = \gamma^*$. Model M2, below, is a generalized version of SUB($\hat{\gamma}$)

which has a modified objective function that penalizes an arc's membership in the set A^*:

$$
\begin{aligned}
\text{M2} \qquad & \max_{x} \; x_{ts} - \sum_{(i,j) \in A^*} x_{ij} \\
\text{s.t.} \; & \sum_j x_{sj} - \sum_j x_{js} - x_{ts} = 0 \\
& \sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\} \\
& \sum_j x_{tj} - \sum_j x_{jt} + x_{ts} = 0 \\
& 0 \le x_{ij} \le u_{ij} \quad \forall\, (i,j) \in A.
\end{aligned}
$$

Proposition 1: *Models M1 and M2 yield equal objective function values: z(M1) = z(M2).*

Although the flow values x will not necessarily be the same in optimal solutions to M1 and M2, this is not important. From the viewpoint of the interdictor, the most significant information to be gained from solving the network interdiction problem is the interdiction sites, $\boldsymbol{\gamma}$.

Proposition 1, as we shall show, facilitates the development of a stochastic network interdiction algorithm.

2. Stochastic Models

In an uncertain environment, a network interdiction attempt may be completely successful, partially successful, or unsuccessful. We consider only the binary case where an attempted interdiction of arc (i,j) is completely successful with probability p_{ij}, and completely unsuccessful with probability $1 - p_{ij}$.

Let I_{ij} be an indicator random variable that is 1 with probability p_{ij} and is 0 with probability $1 - p_{ij}$. The vector of indicator variables $\mathbf{I}$ has state space $\mathcal{I} = \{0,1\}^{|A|}$. For binary, random, interdiction success, the following min-max model is the stochastic analog

of model MINMAX:

$$\text{S - 1} \qquad w^* = \min_{\gamma \in \Gamma} Eh(\gamma, \mathbf{I})$$

where

$$h(\gamma,\mathbf{I}) = \max_{x} \quad x_{ts}$$

$$\text{s.t.} \quad \sum_j x_{sj} - \sum_j x_{js} - x_{ts} = 0$$

$$\sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_j x_{tj} - \sum_j x_{jt} + x_{ts} = 0$$

$$0 \le x_{ij} \le u_{ij}(1 - I_{ij}\gamma_{ij}) \quad \forall\, (i,j) \in A.$$

Now using Proposition 1, S-1 can be reformulated as

$$\text{S - 2} \qquad w^* = \min_{\gamma \in \Gamma} Eg(\gamma, \mathbf{I})$$

where

$$g(\gamma,\mathbf{I}) = \max_{x} \quad x_{ts} - \sum_{(i,j) \in A} I_{ij}\gamma_{ij}x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{sj} - \sum_j x_{js} - x_{ts} = 0$$

$$\sum_j x_{ij} - \sum_j x_{ji} = 0 \quad \forall\, i \in N - \{s,t\}$$

$$\sum_j x_{tj} - \sum_j x_{jt} + x_{ts} = 0$$

$$0 \le x_{ij} \le u_{ij} \quad \forall\, (i,j) \in A.$$

and $g(\gamma,I) = h(\gamma,I)$ for all $(\gamma,I) \in \{0,1\}^{|A|} \times \mathcal{I}$. In this application, the set A^* is defined as $A^* = \{(i,j) \in A \mid \gamma_{ij} = 1,\ I_{ij} = 1\}$. Models S-1 and S-2 allow us to develop upper and lower bounds on w^* with the use of Proposition 2, below.

Proposition 2: *For fixed* γ, $g(\gamma,I)$ *and* $h(\gamma,I)$ *are convex and concave functions on the convex hull of* $\mathcal{I}$, *respectively.*

Letting $\bar{\mathbf{I}}_{ij} = EI_{ij}$ and applying Jensen's inequality twice we obtain

$$g(\gamma,\bar{\mathbf{I}}) \leq Eg(\gamma,\mathbf{I}) = Eh(\gamma,\mathbf{I}) \leq h(\gamma,\bar{\mathbf{I}}).$$

Bounds on w^* may be obtained by solving $\min_{\gamma \in \Gamma} g(\gamma,\bar{\mathbf{I}})$ to obtain $\hat{\gamma}$, producing:

$$g(\hat{\gamma},\bar{\mathbf{I}}) \leq w^* \leq h(\hat{\gamma},\bar{\mathbf{I}}).$$

In the algorithm, we use these bounds extended to a partition of the state space of the indicator random variables. Let $S = \{\mathcal{I}^1,\mathcal{I}^2,\ldots,\mathcal{I}^K\}$ be such a partition of $\mathcal{I}$. Members of S are referred to as "cells." If $\bar{\mathbf{I}}^k = E(\mathbf{I}|\mathbf{I} \in \mathcal{I}^k)$ and $p^k = \text{Prob}(\mathbf{I} \in \mathcal{I}^k)$, then a lower bound on w^* with respect to the partition S is

$$w^* \geq L(S) = \min_{\gamma \in \Gamma} \sum_{k=1}^{K} p^k g(\gamma,\bar{\mathbf{I}}^k).$$

The upper bound associated with a fixed first stage decision vector $\boldsymbol{\gamma}$ is

$$w^* \leq U(S,\gamma) = \sum_{k=1}^{K} p^k h(\gamma,\bar{\mathbf{I}}^k).$$

This upper bound is most easily found by applying the optimal decision $\hat{\gamma}$ obtained during the calculation of $L(S)$. Alternately, it is possible to calculate the least upper bound, with respect to partition S, by solving $\min_{\gamma \in \Gamma} U(S,\gamma)$.

3. Algorithm

The algorithm is a sequential-approximation method that utilizes the bounds developed in Section 2. Its design and implementation are described in more detail in Cormican, Morton, and Wood (1995).

Sequential-Approximation Algorithm

Input: Network G=(N,A), arc capacities u_{ij}, arc interdiction costs r_{ij}, interdiction budget R, special nodes s and t, probabilities of interdiction success p_{ij}, convergence tolerance *toler*.

Output: Interdiction vector γ^* approximately solving S-1, lower bound L^* and upper bound U^* on w^* such that $L^* \leq w^* \leq Eh\,(\gamma^*,\mathbf{I}) \leq U^*$ and $U^* - L^* \leq toler$.

step 1 Let $S = \{\mathcal{J}\}$, $U^* = +\infty$, $L^* = 0$

step 2 Calculate $L^* = L(S)$ and associated minimizer $\hat{\gamma}$;
Calculate $U' = U(S, \hat{\gamma})$

step 3 If $U' < U^*$, let $U^* = U'$ and let $\gamma^* = \hat{\gamma}$

step 4 If $U^* - L^* < toler$, then stop: Approximate solution is γ^*

step 5 Refine the partition S;
Go to step 2.

In this algorithm, step 1 establishes the initial partition and sets bounds. The upper bound is directly determined in step 2, while the lower bound may be found either directly or with Benders decomposition. Step 3 updates the upper bound, if needed, and saves the solution associated with the best upper bound. Step 4 terminates the algorithm if convergence criteria have been met.

The partition refinement of step 5 is a procedure that selects and subdivides a cell from the partition S. The cell for selection is determined by first noting that the gap between the upper and lower bounds is

$$U(S,\hat{\gamma}) - L(S) = \sum_{k=1}^{K} p^k [\, h(\hat{\gamma},\bar{\mathbf{I}}^k) - g(\hat{\gamma},\bar{\mathbf{I}}^k)]$$

with a difference due to each cell of

$$D^k(\hat{\gamma}) = p^k[h(\hat{\gamma},\bar{\mathbf{I}}^k) - g(\hat{\gamma},\bar{\mathbf{I}}^k)].$$

We select a cell k' to subdivide such that $D^{k'}(\hat{\gamma}) = \max_k D^k(\hat{\gamma})$, although other selection criteria could be used.

The refinement of partitions in the selected cell uses a method analogous to that presented by Kall et al. (1988), by conditioning on whether or not an attempted interdiction is successful. Cormican, Morton, and Wood (1995) further explains this and other types of partitioning, as well as more details on the specifics of the algorithm and alternate implementations of various steps of the algorithm.

4. Computational Results

Two cases are tested using OSL on the RISC/6000. The selected network is NS, with $R = (3, 6)$ and all other characteristics as stated in Table 2. In these tests, an interdiction attempt is equally likely to be a success or failure, as $p_{ij} = 0.5 \;\forall (i,j) \in A$. The time results are in Table 15. Since this algorithm does not use GAMS, only the total clock time is listed. The most logical comparison of these times with those of previous techniques is to use the solver times of those methods.

SMALL NETWORK CASE(NS)	NUMBER OF ALGORITHM ITERATIONS	TOTAL CLOCK TIME(hr:min:sec)
$R = 3$	8	00:00:23.6
$R = 6$	31	00:24:35.3

Table 15. Sequential-Approximation Algorithm Summary of Results, Small Network. Solving the network interdiction problem on network NS using the sequential-approximation algorithm, with 6 units of interdiction resource, takes 31 iterations of the algorithm and 24 minutes and 35.3 seconds of solver time.

Although the actual arcs interdicted are not the focus of this research, it is interesting to note that when $R = 3$, one of interdicted arcs was different than those interdicted in the deterministic model. When $R = 6$, again, one interdicted arc was

different.

Comparing the expected value solutions (i.e., all interdictions result in a 50% reduction in flow) is also interesting. This corresponds to the network interdictor measuring the capability of this network to carry flow by solving the problem in which all stochastic parameters are replaced by their expected values. Having done this, he would calculate a maximum flow of 165 for the $R = 3$ case. From Jensen's inequality applied to S-1, this is known to overestimate the true value of expected maximum flow. This true value, solved (optimally, in this case) by the algorithm, is 162.5. For the $R = 6$ example, the numbers to compare are a mean-value problem flow of 180 and an actual flow of 130. The general trend is that as the number of interdiction resources increases, there is an increasing error between these two values. Therefore, the advantages from using this algorithm are best seen when the resource budget is large.

Another benefit of using this algorithm is its adaptability to other forms of uncertainty. Although beyond the scope of this thesis, it is possible to use a variation of this algorithm to analyze stochastic network interdiction problems with arc capacities as discrete random variables, and cases where both interdiction attempts and arc capacities are uncertainties. These additional capabilities are demonstrated in Cormican, Morton, and Wood (1995).

V. CONCLUSION

This chapter describes both the successful and not-so-successful aspects of the models presented in this thesis. Ideas for further model development and computational improvements are noted.

A. GENERAL RESULTS OF MODELS

This thesis has successfully solved the network interdiction problem with models that are applicable to almost any network type and topology. All models are easily generalized to accommodate almost any situation. These models allow:

- non-planar networks,
- any number of source and/or sink nodes,
- any location of source and/or sink nodes,
- directed (only), undirected (only), or both directed and undirected arcs,
- differing costs of interdiction of network components,
- arc and/or node interdictions,
- placing certain arcs and/or nodes "off limits" for interdiction,
- arc capacities as discrete random variables in scenarios, and
- interdiction attempts as independent, probabilistic events.

These features allow much greater flexibility in modelling than most previous methodologies.

The network interdiction problem can be solved effectively using the techniques of Benders decomposition. The possible benefits of using Benders decomposition over direct integer programming include reducing the time required to solve large problems, and the iterative nature of the solution allows the observation of both worst-case and best-case outcomes as the solution progresses. If the solution is only required to be "good," rather

than "the best," the methods in this thesis are ideal.

The appeal of the Benders decomposition algorithm is clear if we think of the network interdiction problem as an adversarial relationship between the network interdictor and the network user. First, the network user maximizes the flow of a commodity in the network. The network interdictor observes this activity, selects interdiction sites, and expends his interdiction budget to minimize the network user's maximum flow. The network user reorganizes in the wake of these interdictions, and again maximizes flow on the remaining network. This process repeats, until neither participant can do any better: an equilibrium has been reached.

The character of this solution results in the intuitively appealing flow dispersion heuristic. Dramatic reductions in the number of Benders decomposition algorithm iterations are observed when the heuristic is used. Three different implementations of the heuristic are tried, with the many-level approximation method seeming to be the most efficient. In one instance, this method decreases the required algorithm iterations by over 80% and lowers the clock time required for a solution by a factor of over 100. In another example, this method outperforms the direct integer programming method by using less than one-third of direct method's solver time.

B. RESULTS FROM MODELS WITH UNCERTAINTY

This thesis is the first work on network interdiction that applies various aspects of uncertainty in a quantitative fashion. In practice, very few network interdiction problems will be free of all uncertainty. Two types of stochastic programming algorithms are analyzed: network scenarios using the L-shaped algorithm, and a more flexible sequential-approximation algorithm. Each algorithm has been used to investigate a different manifestation of uncertainty.

The capacities of arcs and nodes, and the existence (or non-existence) of arcs and nodes are likely candidates for modelling as random variables. The most straightforward approach is to construct a set of scenarios of network capacities and topologies and assign to each scenario an estimated probability of occurrence. This has been successfully accomplished, but the rapidly increasing size of a problem with many scenarios limits the

usefulness of this technique.

The sequential-approximation algorithm avoids the need to limit the number of scenarios. It also allows uncertainty to exist in interdiction success and will soon be adapted to analyze networks with uncertain arc and node capacities. Insights have been gained into the advantages and disadvantages of this approach.

C. POSSIBLE MODEL WEAKNESSES

Having espoused the mathematical programming approach, it must also be recognized that there is a cost to including all the benefits listed in Section A. First of all, in very few cases will an mathematical programming approach be as fast as an approach using algorithms specific to solving network problems, as presented in Phillips (1992), for example. On the other hand, a "network algorithm" will not be as generalizable and adaptable as the models in this thesis.

There are benefits to solving the network interdiction problem using Benders decomposition instead of direct integer programming, but they may be difficult to quantify. The "crossover point," where Benders decomposition becomes the better alternative, is not obvious. In the smaller problems, the integer program will most likely arrive at a solution more quickly than the Benders decomposition algorithm. Other than the compelling evidence in Tables 10, 12, 13 and 14, time did not allow an in-depth study to determine the exact point where the Benders decomposition algorithm becomes preferred over direct integer programming. The next step to such a determination would be to divorce the computations from the time-consuming manipulations of GAMS, since the Benders decomposition model must generate many formulations to solve each network interdiction problem.

The performance of Benders decomposition improves markedly with the inclusion of the flow dispersion heuristic. Unfortunately, the best heuristic technique requires the number of arcs in the network to be increased ten-fold, demanding a large amount of computer memory if the network is large. If the extra memory is not available, the heuristic could be used with a less effective implementation.

Many areas of this research would be supported with further investigation into the

following subjects:

- algorithms implemented in an efficient, compiled, high-level computer language,
- more extensive comparisons of Benders decomposition and direct integer programming methods for large and very large networks,
- decreasing the time needed to solve the Benders master problem,
- quantifying the effect of the flow-dispersion heuristic on algorithm convergence, and
- using simplicial decomposition to implement the flow-dispersion heuristic.

Possible areas of related further research may be:

- incorporating various probability distributions of both arc and node capacity and interdiction success into the sequential-approximation algorithm,
- investigating the effects of time during a prolonged network interdiction campaign,
- investigating the reconstitution of arcs and nodes after each successful interdiction,
- how to obtain the best estimates of arc and node capacities for various applications, and
- applying Benders decomposition techniques to an interdiction problem with more general, non-network constraints.

D. CONCLUSIONS

The models and techniques presented in this thesis could provide effective analytical tools to planners confronted with a network-using adversary. It is hoped that these methods provide enough variety to allow planners to choose the one most suited to their purposes. Whether implemented in GAMS or other high-level computer language, these algorithms are highly transportable and solvable on most computers. If it is desired to solve large-scale network interdiction problems, or problems with a large number of stochastic parameters, the use of a PC becomes somewhat impractical and a more capable

machine should be used to achieve an acceptable solution time.

Although centered on the efforts to stem the flow of illegal drugs into the United States, the possible applications of these techniques are as numerous as networks themselves are numerous.

LIST OF REFERENCES

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B., *Network Flows*, Prentice Hall, 1993.

Bazaraa, M. S., and Shetty, C. M., *Nonlinear Programming: Theory and Algorithms*, John Wiley and Sons, 1979.

Benders, J. F., "Partitioning Procedures for Solving Mixed Variables Programming Problems," *Numerische Methamatik*, Vol. 4, pp. 238-252, 1962.

Brooke, A., Kendrick, D., and Meeraus, A., *GAMS: A User's Guide*, The Scientific Press, 1988.

Cunningham,W., "Optimal Attack and Reinforcement of a Network," *Journal of the ACM*, Vol. 32, No. 3, pp. 549-561, July 1985.

Cormican, K. J., Morton, D. P., and Wood, R. K., "Stochastic Network Interdiction," Working Paper, Naval Postgraduate School, Monterey, California, February, 1995.

Durbin, E. P., "An Interdiction Model of Highway Transportation," Report RM-4945-PR, Rand Corporation, Santa Monica, California, May 1966.

Ford, L. R., Jr., and Fulkerson, D. R., "Maximal Flow through a Network," *Canadian Journal of Mathematics*, pp. 399-404, May 8, 1956.

Ghare, P. M., Montgomery, D. C., and Turner, W. C., "Optimal Interdiction Policy for a Flow Network," *Naval Reserach Logistics Quarterly*, Vol. 18, pp. 37-45, 1971.

Helmbold, R. L., "A Counter Capacity Network Interdiction Model," Report R-611-PR, Rand Corporation, Santa Monica, California, March 1971.

International Business Machines Corp., "Optimization Subroutine Library Guide and Reference, Release 2," document SC23-0519-02, 1991.

Insight, Inc., *XS Mathematical Programming System*, 1994.

Kall, P., Frauendorfer, K., and Ruszczynski, A., Ermoliev, Y., and Wets, R., eds., "Approximating Techniques in Stochastic Programming," *Numerical Techniques for Stochastic Optimization*, Springer-Verlag, 1988.

Kall, P., and Wallace, S. W., *Stochastic Programming*, John Wiley and Sons, 1994.

Lubore, S. H., Ratliff, H. D., and Sicilia, G. T., "Determining the Most Vital Link in a Flow Network," *Naval Research Logistics Quarterly*, Vol. 17, pp. 497-502, December 1971.

McMasters, A. W., and Mustin, T. M., "Optimal Interdiction of a Supply Network," *Naval Research Logistics Quarterly*, Vol. 17, pp. 261-268, September 1970.

Morton, D. P., "A Sampling-Based Algorithm for Two-Stage Stochastic Linear Programming," Working Paper, Naval Postgraduate School, Monterey, CA, 1995.

Phillips, C. A., "The Network Destruction Problem," Report SAND-92-0186C, Sandia National Laboratories, Albuquerque, New Mexico, April 27, 1992.

Steinrauf, R. L., "Network Interdiction Models," Master's Thesis, Naval Postgraduate School, Monterey, California, September 1991.

Sunset Software Technology, *XA Professional Linear Programming System*, 1987.

Van Slyke, R. M., and Wets, R. J.,"L-Shaped Linear Programs with Applications to Optimal Control and Stochastic Programming," *SIAM Journal of Mathematics*, Vol. 17, pp. 638-663, 1969.

Wood, R. K., "Deterministic Network Interdiction," *Mathematical and Computer Modelling*, Vol. 17, No. 2, pp. 1-18, 1993.

Wollmer, R. D., "Removing Arcs from a Network," *Operations Research*, Vol. 12, pp. 807-1076, November-December 1964.

Wollmer, R. D., "Algorithm for Targeting Strikes in a Lines-of-Communication Network," *Operations Research*, Vol. 18, pp. 497-515, May-June 1970.

Wollmer, R. D., "Interception in a Network," *Naval Research Logistics Quarterly*, Vol. 17, No. 2, pp. 207-216, June 1970.

INITIAL DISTRIBUTION LIST

		No. copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5101	2
3.	Professor R. Kevin Wood Department of Operations Research Naval Postgraduate School, Code OR/Wd Monterey, CA 93943-5008	8
4.	Professor David P. Morton Department of Operations Research Naval Postgraduate School, Code OR/Mo Monterey, CA 93943-5008	2
5.	Professor George P. Dantzig Department of Operations Research Stanford University Stanford, CA 94305-4022	1
6.	Chief of Naval Operations (N-81) Navy Department Washington, DC 20350	1
7.	LT Kelly J. Cormican 8863 M-119 Hwy Harbor Springs, MI 49740	2